US 20020178437A1

(19) United States

(12) Patent Application Publication (10) Pub. No.: US 2002/0178437 A1

Blais et al. (43) Pub. Date: Nov. 28, 2002

also 6,505,344 see

(54) OBJECT ORIENTED APPARATUS AND METHOD FOR ALLOCATING OBJECTS ON AN INVOCATION STACK IN A PARTIAL COMPILATION ENVIRONMENT

(75) Inventors: **Marc Noel Blais**, Rochester, MN (US); **Daniel Rodman Hicks**, Byron, MN (US); **William Jon Schmidt**, Rochester, MN (US)

Correspondence Address:
MARTIN & ASSOCIATES, LLC
P O BOX 548
CARTHAGE, MO 64836-0548 (US)

(73) Assignee: **International Business Machines Corporation**, Armonk, NY 10504 (US)

(21) Appl. No.: 09/865,001 IBM

(22) Filed: May 24, 2001

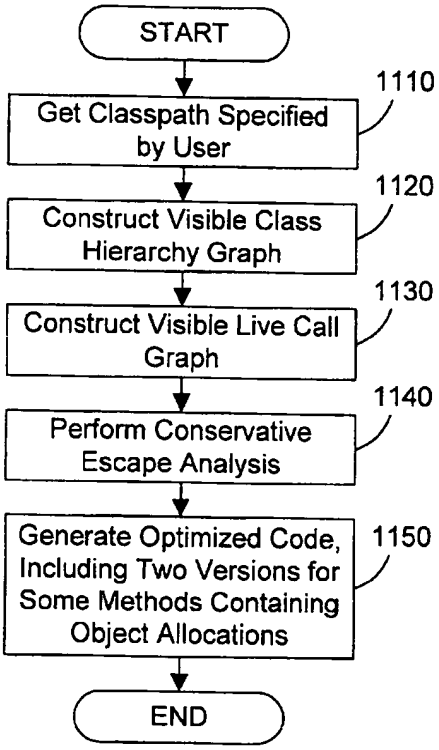
Publication Classification

(51) Int. Cl.[7] G06F 9/45; G06F 9/00

(52) U.S. Cl. 717/140

(57) ABSTRACT

An object oriented mechanism and method allow allocating Java objects on a method's invocation stack in a partial compilation environment under certain conditions. Only the classes that are visible are taken into account when performing escape analysis in accordance with the preferred embodiments. In a first aspect of the invention, conservative assumptions are made to assure that objects are only allocated on an invocation stack when this can be proven safe by examining only those classes in the compilation unit. In a second aspect of the invention, the concept of visible classes is extended to include other classes that may be found from a user-defined classpath that matches the anticipated run-time classpath used to find classes during program execution. When stack allocation decisions for a method depends on such classes that are outside the compilation unit, two versions of run time code for that method are created. One version allocates all objects from the heap, whereas the other allocates some or all objects on the invocation stack. For each class outside the compilation unit that was examined when making stack allocation decisions for a method, information about the requisite class is stored with the method. At run time, the class loader verifies that the run time version of each requisite class matches all stored information about that class. If all information matches for all requisite classes, the version of the method using stack allocation is safe to use at run time. Otherwise, the method using heap allocation must be used.



virtual method call